



ABSTRACT SECTION

COPY OF PAPERS
ORIGINALLY FILED

COPY OF PAPERS
ORIGINALLY FILED

1

2 Abstract 1:

3 This patent application defines computer processes that support using the English
4 language as a computer language. Specifically, methods are defined that let the
5 computer: 1) process input English Language sentences from its memory which are also
6 made up of English Language sentences. 2) use fuzzy logic to allow the computer to
7 respond to differently worded sentences to arrive at the same action i.e., play a card
8 game. Run solitaire. Where (play a game.) also plays the computer game of solitaire; 3)
9 define contexts so that the computers can do: Go to New York. Who is Jim Smith? Go to
10 Washington. Who is Jim Smith? Where answering the question, Who is Jim Smith?
11 depends on the context: Go to New York or Go to Washington; 4) respond to multiple
12 input sentences such as: What is the time. Get my word processor. Go to Greenburg. Who
13 is Rod Smith; 5) where each sentence can be connected to other sentences so that in item
14 4) Get my word processor is connected to: Go to document memory. Get word; 6)
15 process numbers in sentences to be variables or numbers i.e. play the game of 1.222 and
16 What is 3.44 times 22.44; 7) use multi sentence so that one sentence preceding another
17 can change the processing methods for the successor sentence such as: show to 2 decimal
18 places. What is 3.3333 times 2.44467? – shows answer as 8.14; 8) build the computer's
19 memory system from English language text files; 9) talk to each other in English so they
20 can perform parallel computing; 10) use transducers to change voice, vision, touch, other
21 signals to English language so the application can process those signals in its English
22 language memory systems; 11) use any device to convert to the English language; 12) do
23 inference or common sense reasoning to do the following: My car will not start. What

ABSTRACT SECTION

July 26, 2001

Claiming priority to filing a Provisional Patent on 7/29/2000

24 should I do?; 13) use deductive based reasoning to do: My car will not start. What is
25 wrong?; 14) integrate inference based reasoning with deductive based reasoning; 15)
26 Make the computer read English language text files and store those text files in one of its
27 memory systems; 16) make the text file that trained the application in (15) ask the
28 application what it has learning to test that learning; 16) integrate English language data
29 sets associated by sentence memory i.e. Go to English Works memory. Get the
30 enhancement list. If not the enhancement list, then get design notes; 17) be a software
31 agent to itself i.e. the said application calls another said application where the first said
32 application is using the second said application and its specific English language memory
33 systems to get data or cause an action or otherwise respond to the special need of the user
34 in English; 18) conduct all correspondence to users or other devices in English as either
35 imperative or declarative English language sentences; 19) dynamically change any or all
36 sentence memories from external sources – take the existing sentence memories and
37 switching them with new or different sentences memories; 20) define text and non text
38 file memories as English language sentences; 21) make a series of sentences that are
39 being processed to have another set of English Language sentences inserted such that the
40 second set is a subset of the first or inserted somewhere in between the original first set of
41 sentence and the original last set of sentences; 22) do inference based reasoning where
42 the inference creates a new English language sentence such that this new sentence
43 becomes new data for future inferences; 23) store English language sentences such that
44 the same or similar sentence is stored on top of an existing sentence therefore preserving
45 the original; 24) provide a command mode such that the said application just responds to
46 nouns; 25) communicate in English from machine to machine and human to machine

47 and machine to human; 26) utilize the Darwinian mechanism such that sentences created
48 by the said application, independent of those created by the user in the storage of English
49 Language sentences in any of the said applications memories, are created by said
50 application using inferences so that said inferences (created as English Language
51 sentences) can be applied against the existing memory system to see if a match can be
52 found in said memory systems; 27) build English Language sentences by machine that
53 describe events in the computer and can be stored in text files as memory; 28) accept files
54 to reprogram itself and or have new functionality; 29) logically unite disparate activities
55 called from the English Language so that English Language logical analysis can occur to
56 resolve outcomes and make decisions based on the rules of human language and
57 governing norms; 30) oversee human language rule patterns based on other systems in
58 order to analyze those systems and propose new solutions; 31) automatically makes a set
59 of sentences based on one input sentence so the said application can automatically search
60 various memories (made up of English Language sentences) to find a target memory that
61 satisfies the intent of the original sentence; 32) read text stored in text files imported by
62 any means (video, or any transducer input) so that said application can make judgments
63 of said text by said application based on storage of said application memory (made up of
64 English Language sentences); 33) learn from other devices such that the said applications
65 memories in Figure 1 items (5), (8), and (11) are programmed in English Language
66 sentences from outside devices either by human or machines where machines are defined
67 as any non human device.

70

71 Description: Refer to Figure 1. The computer application known as said application (this
72 computer software application) describes methods that allow a computer to process
73 English Language sentences of the type: Declarative, Imperative, X1, and X2. Where
74 input (1) (refer to Figure 1 item (1)) takes in English Language text converted from 1 of
75 N transducers. Text in ASCII format is fed to rule based parsers in Figure 1 item (2) that
76 analyze each input sentence and parse that sentence based, in some cases, on a previous
77 sentence. Said sentence: (What is 3.333 times 2.444?) will display the answer to 2
78 places when previous sentence to (What is 3.333 times 2.444?) is: (display answer to 2
79 places.). Said input sentence (What is 3.333. times 2.444?) finds sentence type in Math
80 memory when user tells the said computer application to: (go to math memory.).
81 Sequence of said sentences to compute: (What is 3.333 times 2.444?) includes the
82 following: (Go to math memory. Display answer 2 places. What is 3.333 times 2.444?).
83 Note: said application will process multiple sentences separated by a period, question
84 mark or exclamation mark.

85

86 Said application is configured to open a sentence memory form (8) or (11) memories.
87 The currently open said memory points to a Math memory as defined by a sentence in the
88 current open memory. User (human) or machine inputs sentence: (go to math memory)
89 and instructs the said computer application to switch to Math memory. Said Math
90 memory contains English Language sentences of type math such an input sentence of
91 type: (What is 4 times 4?) matches (Number times Number) in stored memory sentence:
92 (Number times Number is "This will multiply two numbers."). Said sentence is a

93 declarative sentence containing a noun and prepositional phrase (Number times Number)
94 and a verb phrase: (is “This will multiply two numbers.”) containing the verb: is and the
95 noun phrase contained within the double quotes: (“This will multiply two numbers”).
96 Said sentence verb phrase contains a noun phrase defined with the double quotes even
97 though its literal meaning is not a noun phrase. Said verb phrase is equal to: (is “”).
98 Therefore, said equivalent sentence is: (Number times Number is “”).

100 Said application switches to Math memory from said input sentence: (go to math
101 memory.) (or may be done automatically when said application analyzes input sentence
102 and switches between English Language sentence memories), Next sentence in input
103 sentences instructs said computer application to: (display answer to 2 places.). This
104 sentence is found in common memory at said location (5) and implements the said
105 instruction based on the attached action(s) to this said input sentence.

107 In last sentence of the above 3 input sentences: (What is 3.333 times 2.444?) matches
108 said math memory sentence: (Number times Number is “This will multiply two
109 number.”) and displays the said answer: (The multiplication is 8.15).

111 The said computer application applies input sentences from said transducers show in
112 Figure 1 item (13) to memory systems in Figure 1 items (5), (8), and (11) looking for
113 fuzzy sentence matches which have attached actions for which any action can be an
114 English Language sentences with is fed back into the input at Figure 1 item (13). Said
115 application memory is made up of English Language sentences and said application can

116 have 1 of N memories as defined by a user or copied into the said application memory
117 system from other users at Figure 1 items (5), (8), and (11). A memory is called an
118 object. Said input sentence is directed to 1 of N objects from a previous input sentence
119 coming from an attached action of that sentence as in Figure 1 items (5), (8), and (11) or
120 from the input by a user or machine as in Figure 1 item (1). Said input sentence in Figure
121 1 item (1) or (13) where item (13) is an attached English Language sentence from said
122 previous input sentence finds a fuzzy sentence match in current open object where said
123 open object is selected by a previous input English Language sentence. Said open object
124 containing English Language sentences may be matched by input sentence causing said
125 object's sentence to execute said attached action. Said attached action can be the
126 execution of a computer program, multiple attached English Language sentences or any
127 program function within or outside of said application. Said computer application can
128 send and English Language sentence in Figure 1 items (13) and (14) to another computer
129 running said computer application. In turn, said remote computer application can
130 respond to said computer application English Language sentence by sending back an
131 English Language sentence to said computer application. Said remote computer
132 application may decide to send an English Language sentence(s) to other than first said
133 computer application. Other said computer applications running on other said computers
134 or other devices may eventually send an English Language sentence back to original said
135 computer application.

136
137 Said objects (memory containing English Language sentences) of said computer
138 application can be switched by a said input sentence coming from Figure 1 item(1) or

(13) such that same input can result in different actions depending on said open object. Said objects can contain same or (fuzzy similar) sentences across all objects but with different attached actions. Telling said application to: (go to your New York memory.) and asking: (Who is John Smith?) with said answer: (A person who makes shoes.) vs. telling said application to: (go to your Florida memory.) and asking: (Who is John Smith?) with resulting said answer: (A person who lives at home.). Said application can send same input sentence (polymorphism) to said objects resulting in different results based on attachment to same sentence across all objects.

Said input sentence(s) coming from Figure 1 items (1) or (13) may be used to expose said objects (memories located in Figure 1 items (5), (8), and (11)) where said object is a class for said English Language sentences within said object . For example, directing said computer application to open its vehicle memory object by telling said computer application to: (go to vehicle memory.) exposes the object's sentences to the next input sentence so that for example, next said input sentence coming from Figure 1 items (1) or (13) could be: (What do cars do?). Said object memory (encapsulates) details of vehicles in said object memory. Once said vehicle object memory is open by telling said computer application at Figure 1 items (1) or (13) to: (go to vehicle memory.) said next input sentence at Figure 1 items (1) or (13) is exposed to all of said sentences stored in said vehicle object memory.

Said computer application uses (data abstraction) by telling said computer application at Figure 1 items (1) and (13) to open its object memory. (where item (1) is input by human

162 or other devices which compose an English Language sentence or item (13) where item
163 (13) is an attached sentences stored in Figure 1 items (5), (8), and (11)). Said object
164 memory is open by input sentence and next input sentence is exposed to said open object
165 memory. Said open object memory contains all information on the subject of said object
166 such that any query to said open object memory will contain information relative to the
167 open object's memory. Said sentence: (go to my vehicle memory.) input at Figure 1
168 items (1) or (13) exposes next said input sentence to object memories in Figure 1 items
169 (5), (8), and (11) to a sentence like: (What has 3 wheels?). Where the open memory
170 vehicle object containing information on all vehicles or attached sentences to other object
171 memories can respond to a specific requests for all types of vehicles.

172
173 Said application utilizes (inheritance at the object memory level) when said input
174 sentence to said computer application in Figure 1 items (1) and (13) causes said object
175 memory to open in Figure 1 item (8) and (11). Said open object memory contains a said
176 sentence with attached action that causes a common object memory to open in Figure 1
177 item (5). Said object memory in Figure 1 items (8) and (11) inherits said corresponding
178 common memory in Figure 1 item (5) so that a base class of sentences is combined with
179 other object memories and their sentences shown in Figure 1 items (8) and (11). For
180 example, said input sentence (go to your vehicle memory) in Figure 1 item (1) and (13)
181 causes said object memory to open in Figure 1 item (8) or (11). Said open object
182 memory contains a sentence with an attached sentence (Figure 1 item (13)) to open 1 or
183 N said common memories in Figure 1 item (5). Both object memory sentences from
184 Figure 1 item (8) or (11) and item (5) are stored in computer RAM. User now inputs next

185 input sentence at Figure 1 item (1) or (13) to expose open object memories Figure 1 item
186 (5) and item (8) or Figure 1 item (5) and item (11). The vehicle open object memory in
187 Figure 1 items (8) and (11), for example, will process input sentences at Figure 1 items
188 (1) and (13) about vehicle attributes for that open object memory, but it will also process
189 input sentences regarding 2 and 3 wheeled vehicles based on sentences stored in the
190 selected common object memory. Therefore, user inputs: (go to your vehicle memory.)
191 and said computer application opens vehicle memory and then opens the related common
192 memory that holds sentences and actions regarding 2 and 3 wheeled vehicles. In this
193 regard the open vehicle memory in Figure 1 item (8) or (11) inherits vehicle base class
194 information from the corresponding common memory object in Figure 1 item (5).
195 Typical input sentences would be as follows: (open your vehicle memory. What do cars
196 do? What has 2 wheels?). Where: (What has 2 wheels?) comes from sentences in the
197 common object memory (Figure 1 item (5) and opened by the sentence: (What do cars
198 do?) located in object memories in Figure 1 items (8) and (11).

199
200 Said application utilizes (inheritance at the sentence level) when said application process
201 English Language sentences to do inference based reasoning. Said example sentences:
202 (My car will not start. What should I do? vs. What is wrong? For deductive based
203 reasoning) causes said application to answer: (take a taxi or other derived answer
204 extracted from existing said application memory). Said application analyzes input
205 sentences: (My car will not start. What should I do?) and reverses the order so that said
206 input sentences become: (What should I do? My car will not start.) where said sentence:
207 (What should I do?) causes common memory in Figure 1 item (5) to set up logic in

208 Figure 1 item (2) so that said sentence: (My car will not start.) is processed using
209 inference based reasoning. Attached action of the sentence (My car will not start.) stored
210 in said application memory in Figure 1 items (5), (8), and (11) is (Vehicles transport
211 things. Go to vehicle memory.) where said application goes to its vehicle memory
212 knowing its looking for an inference from the said application based on input sentence:
213 (What should I do?). Said application process attached sentences: (Vehicles transport
214 things. Go to vehicle memory.) which goes to input in Figure 1 item (13) from (My car
215 will not start.) isolating verb (transport) and applying a search in vehicle memory looking
216 for same said verb (transport). Verb (transport) in said originating sentence with attached
217 sentence: (Vehicles transport things.) also has a hierarchical number associated with the
218 sentence: (Vehicles transport things.) of 1. In said target memory: (Go to vehicle
219 memory.) searched verb (transport) at hierarchical number 1 is found in target memory
220 with said sentence: (Cars transport people.) with attached inference: (Take a taxi.). Said
221 application displays said inference solution to user (Take a taxi.) based on input
222 sentences: (My car will not start. What should I do?).

223

224 The said application has many functions all of which can be implemented by inputting
225 English Language sentences of the type declarative, imperative, interrogative, and
226 exclamatory. To put the said application in the learn mode, a sentence like: (please
227 learn.) and others using fuzzy logic (stored English Language sentences etc.) will store
228 English language sentences in a declarative format. (Note that fuzzy logic is defined as
229 storing an English Language declarative (factual) sentence in 1 of N memories shown in
230 Figure 1 in items (5), (8), and (11) such that the stored (learned sentence) defines like

231 words by using an English Language thesaurus). All learned sentences will be stored in 1
232 of N computer memories at locations (5), (8), and (11) in figure 1. Teaching the said
233 application to learn how to play the computer game of solitaire can be done by storing a
234 declarative sentence like: (card game of solitaire is "Playing a solitaire card game.") and
235 attaching the solitaire program executable (sol.exe) to the said declarative sentence. Any
236 number of attached actions can be stored with a learned declarative English Language
237 sentence including other English Language sentences that are fed back into the input at
238 Figure 1 item (13) or to the output item (14). From the stored sentence: (card game of
239 solitaire is "Playing a solitaire card game.") any declarative, imperative, interrogative,
240 and exclamatory sentence from machine to human can input an ASCII text sentence to
241 play a solitaire card game using said application. Some of the word combinations
242 include: (please play cards, get solitaire, play a game, load a card game, Can you play
243 solitaire?) etc. These word combinations are formed using English Language sentence
244 rules and implemented by humans or machines and input into said application in Figure
245 1 item (1) or item (13) to match words in object memory in Figure 1 items (5), (8), and
246 (11). Said input sentences finds object memory sentence in Figure 1 items (5), (8), and
247 (11) which has been learned and stored by user in declarative English Language format as
248 in: (card game of solitaire is "Playing a solitaire card game."). Each of the input
249 sentences (declarative, imperative, interrogative, and exclamatory) will match various
250 words of the learned/stored declarative English Language sentence to cause said
251 computer application to play the computer game of solitaire. The method described
252 above in Figure 1 item (5) and (8) learns declarative English Language sentences in
253 Native mode and stores those sentences in 1 of N user object memories stored in

254 computer files. File memories are defined by users and contain English Language
255 declarative sentences and associated stored actions as defined by the user. A stored
256 action is any event or another English Language sentence that may be attached to a stored
257 declarative sentence. Said stored actions can instruct said application to switch to a new
258 object memory file for which the next input English Language sentence from said
259 transducers in Figure 1 item (1) can cause new functionality to occur. Likewise said
260 application can use text files (shown in Figure 1 item (11)) that are English Language
261 declarative sentences separated into paragraphs as shown in Figure 2.

262

263

264

265 1. play a card solitaire game.

266 do: (sol.exe, c:\Windows).

267 display: Playing the card game of solitaire.

268

269 2. play a board chess game.

270 do: (chess.exe, c:\Windows).

271 display: Looking at my Files.

272

273 3. get my your this text paragraph file.

274 do: (notepad.exe, \Computer text memory\game text memory.txt).

275

276 4. show my computer files folders.

277 do: (Explorer.exe, c:\).

278

279 3x. show what the Ews your architecture diagram lay out layout.

280 do: (c:\Program Files\PowerPointViewer\Ppview32.exe,c:\MyPro386w\PwrPt\diagram3).

281

282

283 3y. let Something destroy the bark.

284 display: going to tree memory.

285 do: go to tree memory.

286

287

288 4. who is Bill William Jefferson Clinton?

289 do: play cards.

290 do: show my computer files.

291 display: A guy who lives in New York.

292 related: His mother lives in Colorado.

293

294

295 5. who is Hillary Hilary Rodam Clinton?

296 display: Wife of the President of the United States.

297 related: Hillary's parents.

298 do: search. play cards.

299

300

301 6. get me picture map of England Great Britain.

302 do: (president.bmp, c:\MyPro386w\bitmap files\uk.bmp).

303 display: Showing a map of England.

304

305

306 7. find a chair car or boat.

307 do: play cards. go to core.

308

309

310 8. show my C disk drive space availability capacity amount on my C drive.

311 do: (Drvspace.exe, c:\).

312 display: Showing the c drive space.

313

314

315 8a. show my A disk drive space availability capacity amount on my C drive.

316 do: (Drvspace.exe, a:\).

317 display: Showing the a drive space.

318

319

320 8b. do disk clean or defragmentation.

321 do: (defrag.exe, c:\Windows).

322 display: defrag the selected disk drive.

323

324

325 9. who owns or is the owner of the Chicago and Northwestern?

326 do: play solitaire.

327

328

329 10. play a chess board game.

330 display: playing chess.

331

332

333 11. call a Person.

334 do: If get Person's phone number. then call Person. else call information.

335 display: sentence logic test. Also, Chuck's phone number should be asserted into

336 memory which would allow call Chuck to happen.

337

338

339 12. call 411 information.

340 do: (sol.exe, c:\Windows).

341

342

343 12a. what does Intel PC notes makes computers easy to use say.

344 do: (Intel ease of use.bmp, c:\MyPro386w\bitmap files\Intel ease of use.bmp).

345 display: From Intel's web site on 12/15/98.

346

347

348 13. He was in New York getting his car.

349 display: getting his car in New York.

350

351

352 14. This is a Chuck test.

353 display: This is a Chuck test.

354

355

356 15. get calculator adder my math machine.

357 do: (calc.exe, c:\Windows).

358 display: can also be made into a scientific calculator.

359

360

361 16. something destroyed the bark.

362 do: play cards.

363

364

365 17. show my design note additions.

366 do: (notepad.exe, a:\other\design notes1).

367 display: These are the design notes on the ewe diskette.

368

369

370 18. show the my problems issues list.

371 do: (notepad.exe, a:\Other\ews problem list).

372 display: Ews problem list.

373

374

375 19. go get the other next alternative Bill.

376 do: get test 1. Who is Bill. get test 2.

377 display: Getting information on the other Bill.

378

379

380 20. something destroyed the bark.

381 do: play cards.

382

383

384 21. get the new proposed functions functional capabilities notes list.

385 do: (notepad.exe, a:\Other\function list).

386

387

388 22. set up Ews development environment platforms software tools.

389 do: do Dos. show problem list. show functional notes. show design notes.

390

391

392 Said paragraphs of English Language declarative sentences can be matched by an input
393 English Language sentence from machine to human of type declarative, imperative,
394 interrogative, and exclamatory. Matching the input English Language sentence of type
395 declarative, imperative, interrogative, and exclamatory with English Language
396 declarative sentences stored in each text paragraph as shown in Figure 2 such that one
397 paragraph with attached actions for which one action can be made up of an English
398 Language sentence will call another paragraph sentence who's attached action could call
399 Native memory which switches to a new text file shown in Figure 1 using mechanisms in
400 items (5), (8), and (11). Text file memory switching will enable the following: (Go to
401 Washington, D.C. What is Mr. Smith's phone number? Go to New York. What is Mr.
402 Smith's phone number?). Where the sentences (Go to Washington, D.C. and Go to New
403 York.) switch memories giving the said application through parsers in Figure 1 item (2)
404 the ability to point to a new set of English Language memories defined in Figure 1 items
405 (5), (8), and (11).

406

407 Said application can read a text files made of up of English Language sentences found by
408 telling said computer application to go out and read said text file such that read text file
409 teaches said application by transferring English Language sentences from said text files
410 into said application in Figure 1 item (1). Once said application has transferred English
411 Language sentences into said application memory object in Figure 1 item (8), said
412 application continues to read said text file which further instructs said application, in
413 English Language, to do what said computer application has learned by storing English
414 Language sentences in said computer application object memory in Figure 1 item (8).

415 Said text file that teaches said computer application is shown in Figure 3. Said
416 application is told, using an English Language sentence in Figure 1 item (1) or (13) to
417 learn from a text file when user instructs said computer application to go out and read
418 said text file. The said text file that teaches the said computer application to play the
419 computer game of solitaire is as follows:

420

421 Go to your learning memory. Save this data. "sol exe".

422 Save the user sentence. "solitaire card game" is "playing solitaire".

423 Stop learning text sentences. play solitaire. Go to your user memory.

424

425 Figure 3

426

427

428 Said common memory in Figure 1 item (5) contains English Language declarative
429 sentences stored in ASCII text such that stored English Language sentence in common
430 memory is matched with an input English Language input sentence of the form
431 declarative, imperative, interrogative, and exclamatory located in Figure 1 item(1) or (13)
432 from machine or human. When said match is made in common object memory with
433 input sentence, search of subsequent memories in Figure 1 items (8) and (11) is
434 prevented. If search of common memory fails to find a stored English Language
435 sentence that matches input sentence using fuzzy logic, search continues in Figure 1 item
436 (8) and then to item (11). Said application in Native memory in Figure 1 item (8) or item
437 (11) can have a stored English Language sentence attached to a stored action that

switches common memory in Figure 1 item (6). Said common memory switches to a new set of English Language sentences which may become aligned to the context of said Native memory in Figure 1 item (8) and item (11) as defined by the user when said application is in learn mode storing actions with said memories in Figure 1 item (8) and (11).

Said common memory in Figure 1 item (5) enables the following functionality to occur in said application using deductive and inference based reasoning. In deductive based reasoning input English Language sentences show in Figure 1 item (1) may include a stream of declarative sentences such as: (My car will not start. There are no headlights. What is wrong?). Where said sentence: (What is wrong?) resides in common memory and causes deductive based processing of said two input sentences: (My car will not start. There are no headlights.). Said common memory sentence stored in Figure 1 item (5) uses fuzzy logic such that variation of said sentence: (What is wrong?) can include: (get a solution. What is it?) where these variations are connected to the same action causing said deductive solution to the English Language input sentences in Figure 1 item (1): (My car will not start. There are no headlights.). Said common memory sentences (What is wrong? etc.) have stored actions attached to said sentences as shown in Figure 1 item (5) causing the appropriate process to occur when said sentences (My car will not start. There are no headlights.) are processed in memories in Figure 1 items (8) and (11).

Said declarative input sentences for deductive based reasoning are fired into computer memory based on matches found in Native and text memory at locations Figure 1, items

(8) and (11). Stored actions associated with each declarative input sentence define a confidence factor when all input declarative input sentences match all stored declarative sentences shown in Figure 1 items (8) and (11). Associated stored actions found with matching input sentences and stored declarative sentences found in items (8) and (11) cause an expected result which could include sending a new English Language sentence to the input located in Figure 1 item (13). The stored actions may contain a number of sentences the are either fed back to the input in at Figure 1 item (13) or to external computers, humans, or appliances in Figure 1 item (14). Where an appliance accepts an English Language sentence, reacts to that sentence, and sends an English Language sentence back to the sending device as in Figure 1 item (1) or other devices in a network. All generated English Language sentences resulting from memory transactions whether generated from a stored action or composed using internal parsing technology in Figure 1 item (2) interact with the memory of said computer application or the memories of other devices creating a dialog using English Language sentences between devices. Any decoded English Language sentence results in the activation of a stored action. A simple device may decode specific English Language sentences (decoded to ASCII text for processing) sent from said application resulting in said device sending back an English Language sentence in response to said application or other said applications or other devices or humans either as text, voice or other electrical signals represented by said English Language sentence(s). Said stored actions result in new sentences or memory switching as shown in Figure 1 items (4), (6), (7), (9), and (10) from said application or canned sentences from appliances causing said work to be done for user.

484 When said application is not in the learn mode, English Language declarative input
485 sentences shown in Figure 1 item (1) find stored declarative sentences in Figure 1 item
486 (8) and item (11). When found, those declarative sentences are connected to stored
487 actions which define the solution for deductive based reasoning. Associated solutions
488 include English Language sentences that may be fed back to the input Figure 1 item (13)
489 or go outside of the said computer application shown at Figure 1 item (14) to other
490 devices. Those devices can be said computer application running on other computers or
491 appliances. In all cases, outside sources, either human or machine, communicate to each
492 other in English Language sentences.

493

494 Said computer application can integrate deductive based reasoning with inference based
495 reasoning. Deductive based reasoning is the storage of English Language sentences
496 connected together within and across object memories. Connecting sentences and object
497 memory together occurs in the learn mode as user defines connecting relationships
498 between sentences. User typically enters declarative sentences in Figure 1 items (1) and
499 (13) which map into learned stored connected sentences in object memory in Figure 1
500 items (5), (8), and (11). A typically example session could include the following input
501 sentences: (go to vehicle memory. My car will not start. There is no dome light. What is
502 wrong?). When said computer application receives first said sentence: (go to vehicle
503 memory.) said application switches to vehicle object memory in Figure 1 items (8) and
504 (11). Once switched, said second sentence: (My car will not start.) finds a sentence
505 match in current open object vehicle memory. Matched said sentence (My car will not
506 start.) has an attached sentence: (go to start problem memory.) Said computer application

507 builds new remaining sentences: (go to start problem memory. There is no dome light.
508 What is wrong?) and said computer application switches to: (go to start problem
509 memory) in Figure 1 item (13) causing new object memory to open in Figure 1 items (8)
510 and (11). Next input sentence of remaining sentences: (There is no dome light. What is
511 wrong?) finds matching sentence in current open object memory (go to start problem
512 memory.) in Figure 1 items (8) and (11). As said matching sentences are found
513 throughout object memories, said attached actions to each matching sentence is stored in
514 said computer application RAM area noting the number of matched sentences compared
515 to the number of matched sentence with a group as defined by the user in said application
516 learn mode. Said last sentence: (What is wrong?) causes said application to find said
517 sentence in object common memory in Figure 1 item (5). Said sentence: (What is
518 wrong?) has attached actions that causes said application to process data stored in said
519 computer application RAM resulting form said sentence matches found in said object
520 memory in Figure 1 items (8) and (11). Resulting said solution: (The car battery may be
521 dead or disconnected. Confidence factor 100 percent). Said data in Ram may also
522 include sentences: (go to battery memory. battery provides power.) such that when user
523 inputs: (What should I do? – asking said application to make an inference or alternative
524 to using a battery, and where said sentence: (What should I do?) exists in said common
525 object memory in Figure 1 item (5)) said application transfers said sentences: (go to
526 battery memory. Battery provides power.) to Figure 1 item (13). Said inference
527 sentences: (go to battery memory. Battery provides power.) switch to battery memory as
528 directed by input sentence: (go to battery memory.) in Figure 1 items (8) and (11). Next
529 said input sentence: (Battery provides power.) finds said sentence in battery object

530 memory in Figure 1 items (8) and (11). Said sentence: (Battery provides power.) has
531 attached sentence: (go to electrical power memory.) causing said application to switch
532 object memory in Figure 1 items (8) and (11). Said application isolates verb (provides)
533 and searches through object memory: (go to electrical power memory.) for said verb
534 (provides) at same said hierarchical level of 2 as defined by said previous sentence:
535 (Battery provides power.). Said computer application finds sentence in said open object
536 memory (go to electrical power memory.) in Figure 1 items (8) and (11) finds sentence:
537 (Devices provide electrical power.). Said found sentence: (Devices provide electrical
538 power.) at hierarchical level 2 has attached user comment: (Replace or recharge battery.).

539

540 Said application uses multi sentence technology to: 1) open new object memory; 2) find
541 matches in object memory in Figure 1 items (5), (8), and (11) as defined by input from
542 Figure 1 items (1) and (13); 3) cause said found sentence to execute any action as
543 defined by said attached computer programs; 4) send attached sentences found in
544 memory at Figure 1 items (5), (8), and (11) to output in Figure 1 items (14) to said other
545 computers using said application or devices responding to said English Language
546 sentences or their said symbolic derivatives; 5) enable parallel computing where said
547 computer or human dialog in English Language is sustained using said application within
548 or across computer platforms is complete and dialog stops such that all sentences have
549 competed resulting in work being done by said application. Said found sentences in
550 Figure 1 items (5), (8), and (11) apply new sentences to input at Figure 1 item (13). Input
551 sentences at Figure 1 item (1) or item (13) can be from 1 to N sentences where sentences
552 from Figure 1 item (13) can be attached to stored sentences in Figure 1 items (5), (8), and

(11). Input sentences in Figure 1 item (1) can be composed by human or machine in any number from 1 to N.

Said application logically processes input sentences such that the following example input sentences coming from Figure 1 item (1) or (13) from deposited sentences in Figure 1 items (5), (8), and (11) are: (please play cards. If you played cards then show card instructions. Otherwise, play a game of chess.) Where said application finds match in said object memory in Figure 1 items (5), (8) and (11) for said sentence: (please play cards.) noting that if said match did not occur from input in Figure 1 items (1) or (13) and said open object memory in Figure 1 items (5), (8), and (11) that said application generates internal message: (my memory is blank.). When said message is generated (my memory is blank.) because said application could not find match to said input sentence: (please play cards.) to said memory, said next input sentence: (If you played cards then show card instructions.) is disregarded such that the sentence: (Otherwise, play a game of chess.) is processed through said application memory in Figure 1 items (5), (8) and (11) resulting said application playing the computer game of chess. Said application with said stored English Language sentences stored in said object memories in Figure 1 items (5), (8), (11) have attached sentences that can be sent to said application output in Figure 1 item (14) or display on said application user screen or put into ASCII text to voice transducer such that when said input sentences matches said memory sentences with said attached action which could be a said English Language sentences, that said sentence could be announced to a local or remote speaker.

576 Said application can be programmed in ordinary English Language sentences by storing
577 said sentences in said object memories in Figure 1 items (5), (8), and (11). Format of
578 said stored English Language sentences can be generated by computer program and auto
579 loaded into said application object memory. Said application can request auto load of
580 new said application object memories to adapt to new requirements as defined by an
581 English Language sentence form human or machine.

582

583 Said application uses two methods to sort through user object memory in Figure 1 items
584 (5), (8), and (11). Method 1 indexes said object memory in Figure 1 item (8) to open
585 object memories in Figure 1 item (11) with single sentence input or multi sentence input.
586 Method 2 builds internal sentences to reapply those sentences in Figure 1 item (13) in
587 order to sort through object memory in Figure 1 items (5), (8), and (11). In Method 1,
588 human or machine enters sentence(s) in Figure 1 item (1) or said application enters
589 sentence(s) in Figure 1 item (13). Said object memory switches object memory based on
590 attached sentence associated with matched sentence in Figure 1 items (5), (8), and (11).
591 For example, the input sentence(s): (Go to New York memory. Who is John Smith?)
592 causes said index sentence: (Go to New York memory.) to switch to New York object
593 memory when said application is told with an English Language sentence to: (please go
594 into search mode.). (Go to New York memory.) is in an index of all object memories
595 located in Figure 1 item (8). Attached action of said found sentence (Go to New York
596 memory.) switches object memory to New York memory in Figure 1 items (8) or item
597 (11). Said next sentence locates said sentence (Who is John Smith?) through existing
598 index or in memories in Figure 1 items (11) or items (5). Said open object memory is

599 open containing 1 of N sentences of which one is fuzzy similar to (Who is John Smith?).
600 Said next sentence: (Who is John Smith?) searches for a match in said open object
601 memory in Figure 1 items (8), and (11) and displays answer to (Who is John Smith?) as
602 defined in said computer application learn mode. In said learn mode, user would have
603 previously stored and defined a sentence like: (“John Smith New York friend” is “John
604 Smith is a friend from New York.”) whereby the input sentence: (Who is John Smith?)
605 in Figure 1 items (1) or (13) would look for a match in Figure 1 items (5), (8), and (11)
606 such that the input sentence is parsed (broken apart according to the rules: sentence =
607 noun phrase + verb phrase) in Figure 1 item (2) or their derivatives and applied against
608 the stored sentence: (Who is John Smith?) found in Figure 1 items (5), (8), and (11).
609 Found said sentence: (Who is John Smith?) may have 1 of N stored actions. Said stored
610 actions can include any combination of English Language sentences and other data types
611 depending on user configuration of stored sentence in Figure 1 items (5), (8), and (11).
612 Said stored actions are attached to said stored sentence in said computer application using
613 said computer application learn mode. Said stored sentences and said stored actions are
614 stored in Native, text or SQL data types as shown in Figure 1 items (5), (8), and (11).